FIG.1

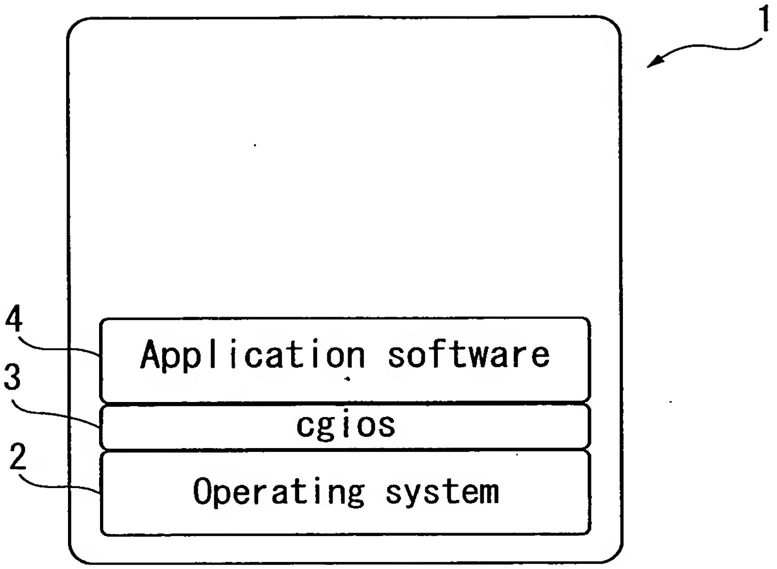

1
4
3
2
Application software
cgios
Operating system

FIG.2

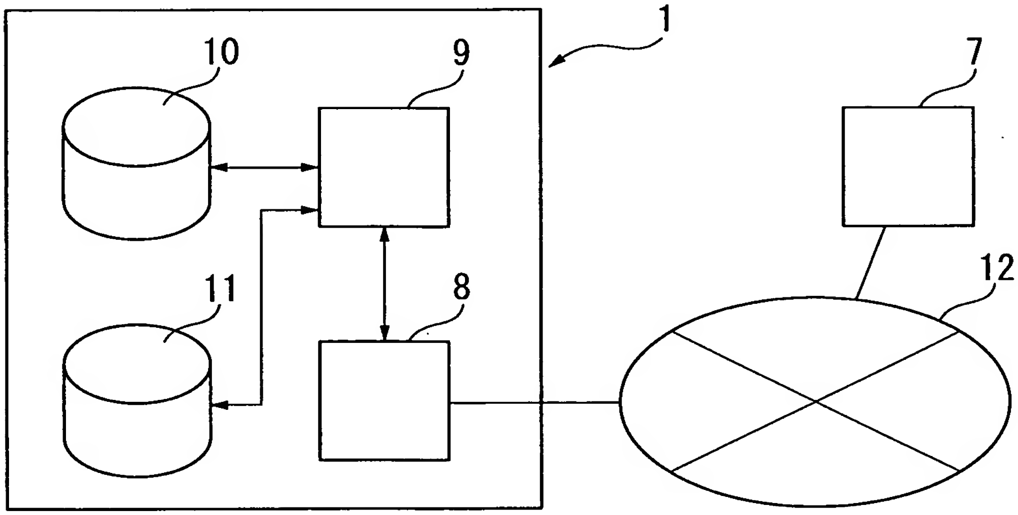
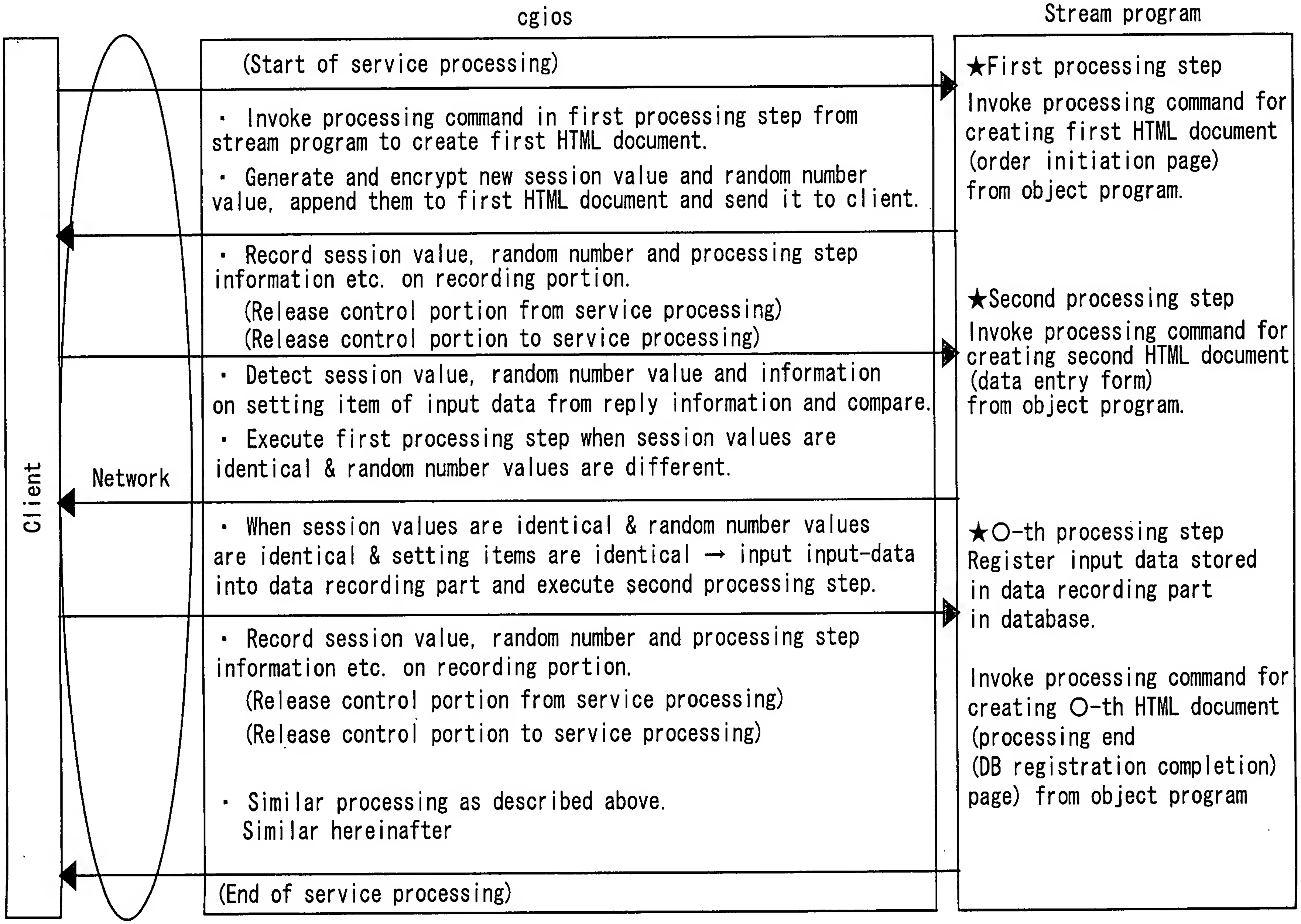

1
10
9
7
11
8
12

FIG.3



(Start of service processing)

- Invoke processing command in first processing step from stream program to create first HTML document.
- Generate and encrypt new session value and random number value, append them to first HTML document and send it to client.

- Record session value, random number and processing step information etc. on recording portion.
  (Release control portion from service processing)
  (Release control portion to service processing)

- Detect session value, random number value and information on setting item of input data from reply information and compare.
- Execute first processing step when session values are identical & random number values are different.

- When session values are identical & random number values are identical & setting items are identical → input input-data into data recording part and execute second processing step.

- Record session value, random number and processing step information etc. on recording portion.
  (Release control portion from service processing)
  (Release control portion to service processing)

- Similar processing as described above.
  Similar hereinafter

(End of service processing)

★First processing step
Invoke processing command for creating first HTML document (order initiation page) from object program.

★Second processing step
Invoke processing command for creating second HTML document (data entry form) from object program.

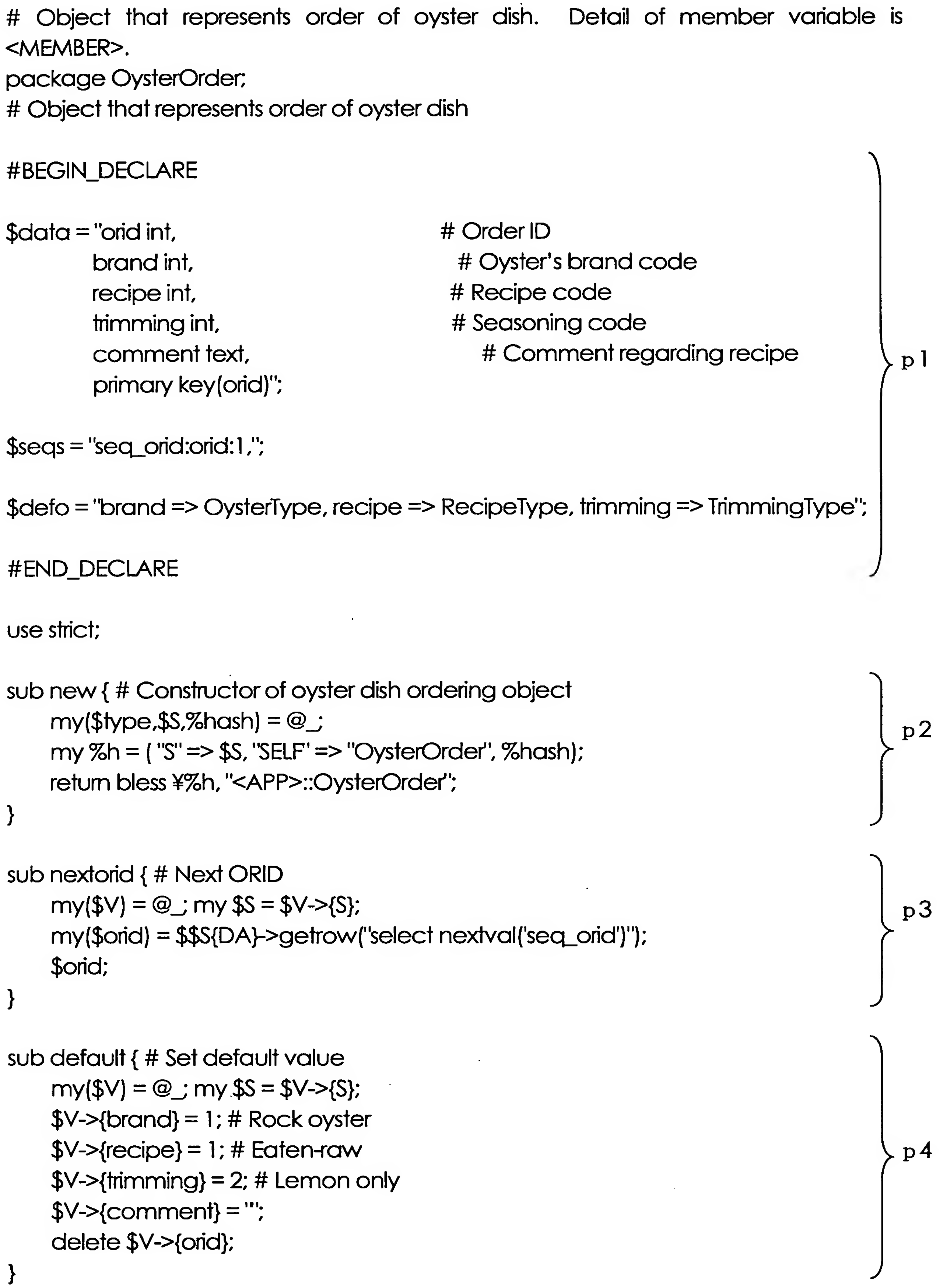
★O-th processing step
Register input data stored in data recording part in database.

Invoke processing command for creating O-th HTML document (processing end (DB registration completion) page) from object program

FIG.4

```
ustream input_order($o) { # order entry stream
  top:
    $o = new <APP>::OysterOrder($S);
    $o->default;

  first:
    page(input_order.page) {
        $V->{SLOT} = { slot => $o->input };
    }
    if ($V->{action} eq "next") { goto second; }
    goto first;

  second:
    if ($o->{ERROR} = $o->check) { goto first; }
    page(print_order_conf.page) {
        $V->{SLOT} = { slot => $o->print };
    }
    if ($V->{action} eq "back") { goto first; }
    elsif ($V->{action} eq "next") { goto third; }
    goto second;

  third:
    $o->{orid} = $o->nextorid;
    $$S{DA}->saveObject($o);

  fourth:
    page(print_order_done.page) {
        $V->{SLOT} = { slot => $o->print };
    }
    if ($V->{action} eq "next") { goto first; }
    goto fourth;
}
```

FIG.5



```
# Object that represents order of oyster dish.  Detail of member variable is
<MEMBER>.
package OysterOrder;
# Object that represents order of oyster dish

#BEGIN_DECLARE

$data = "orid int,                          # Order ID
         brand int,                           # Oyster's brand code
         recipe int,                         # Recipe code
         trimming int,                       # Seasoning code
         comment text,                          # Comment regarding recipe
         primary key(orid)";

$seqs = "seq_orid:orid:1,";

$defo = "brand => OysterType, recipe => RecipeType, trimming => TrimmingType";

#END_DECLARE

use strict;

sub new { # Constructor of oyster dish ordering object
     my($type,$S,%hash) = @_;
     my %h = ( "S" => $S, "SELF" => "OysterOrder", %hash);
     return bless ¥%h, "<APP>::OysterOrder";
}

sub nextorid { # Next ORID
     my($V) = @_; my $S = $V->{S};
     my($orid) = $$S{DA}->getrow("select nextval('seq_orid')");
     $orid;
}

sub default { # Set default value
     my($V) = @_; my $S = $V->{S};
     $V->{brand} = 1; # Rock oyster
     $V->{recipe} = 1; # Eaten-raw
     $V->{trimming} = 2; # Lemon only
     $V->{comment} = "";
     delete $V->{orid};
}
```

FIG.6

```
sub input { # Order entry form
     my($V) = @_; my $S = $V->{S};
     print_webform( # Order entry form
'<table>
 <ifdef ERROR><span class=error><!ERROR></span></ifdef>
 <tr><td> Brand of oyster</td><td>
  <input type=select name=brand value=$brand choice=[OysterType]></td></tr>
 <tr><td> Recipe of oyster </td><td>
  <input type=select name=recipe value=$recipe choice=[RecipeType]></td></tr>
 <tr><td> Trimmed seasoning</td><td>
  <input type=TrimmingType name=trimming value=$trimming></td></tr>
 <tr><td> Comment</td><td>
  <input type=text name=comment value=$comment size=60></td></tr>
</table>');
}
```

p5

```
sub check {
     my($V) = @_; my $S = $V->{S};
     my $count = 0;
     my $value = $V->{trimming};
     while ($value>0) { $count += $value & 1; $value >>= 1; }
     my $error = "";
     if ($count >= 3) {
          $error = "<li> You can only attach up to two kinds of seasoning at once.";
     }
     $error;
}
```

p6

```
sub print { # Fair copy of order in HTML format
     my($V) = @_; my $S = $V->{S};
     print_webtext( # Fair copy of order in HTML format
'<table>
 <tr><td> Brand of oyster </td><td><%brand></td></tr>
 <tr><td> Recipe of oyster</td><td><%recipe></td></tr>
 <tr><td> Trimmed seasoning</td><td><%trimming></td></tr>
 <tr><td> Comment</td><td><%comment></td></tr>
</table>');
}
```

p7

FIG.7

(a) <Page for inputting order:/main/input_order.page>

```
        <h3> Please select brand of oyster and recipe</h3>
        <#slot>
        <br>
    <input type=submit action="next" label=" NEXT '>
```

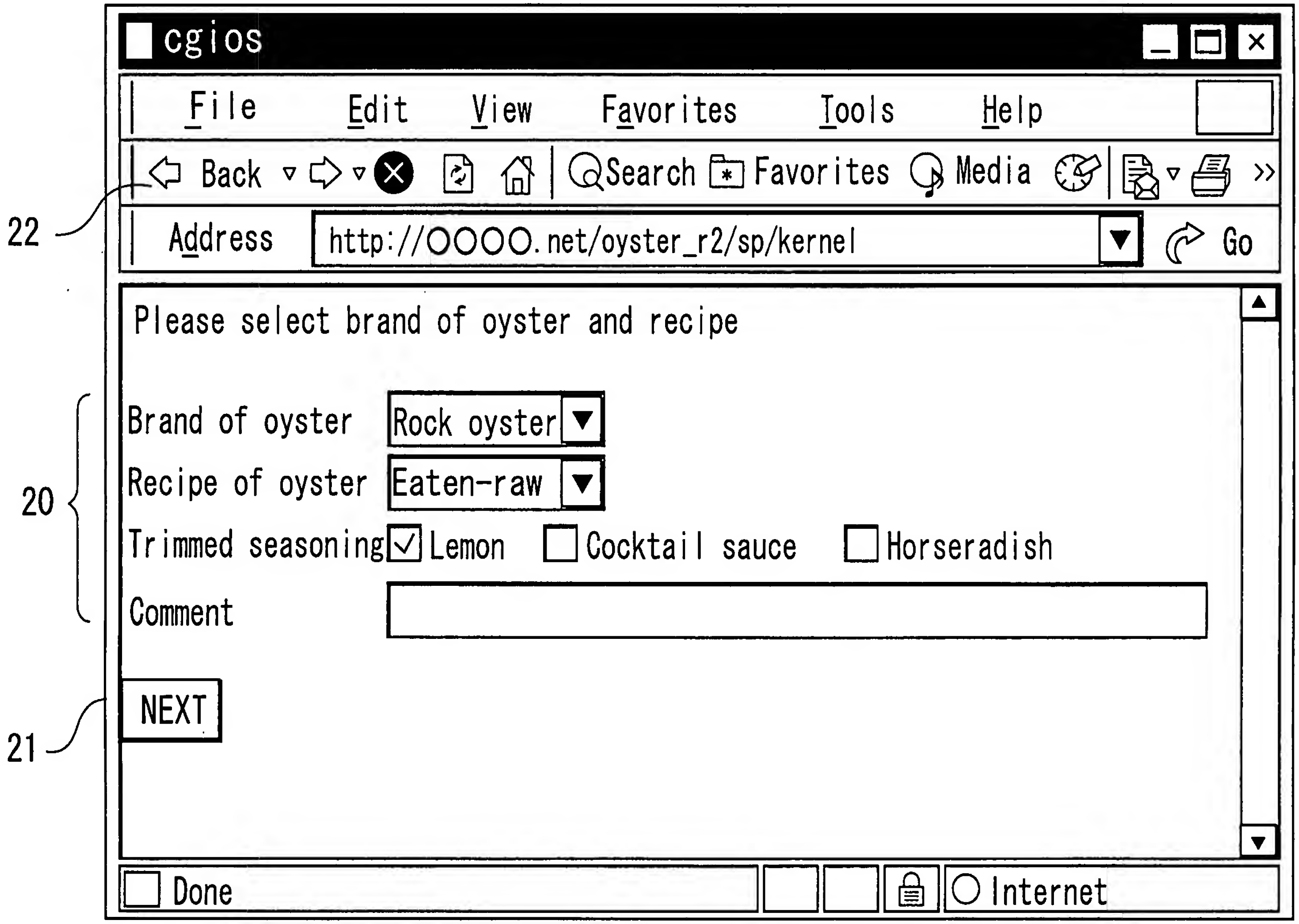
(b) <Page for confirming order:/main/print_order_conf.page>

```
        <h3> Please confirm your order.</h3>
        <#slot>
        <br>
        <input type=submit action="next" label=" ORDER EXECUTE'>
<input type=submit action="back" label=" BACK'>
```

(c) <Page for notifying that order is completed.:/main/print_order_done.page>

```
        <h3> The following order has been accepted.</h3>
        <#slot>
        <br>
<input type=submit action="next" label=" BACK TO BEGINNING'>
```

FIG.8



http://OOOO.net/oyster_r2/sp/kernel

Please select brand of oyster and recipe

Brand of oyster: Rock oyster ▼

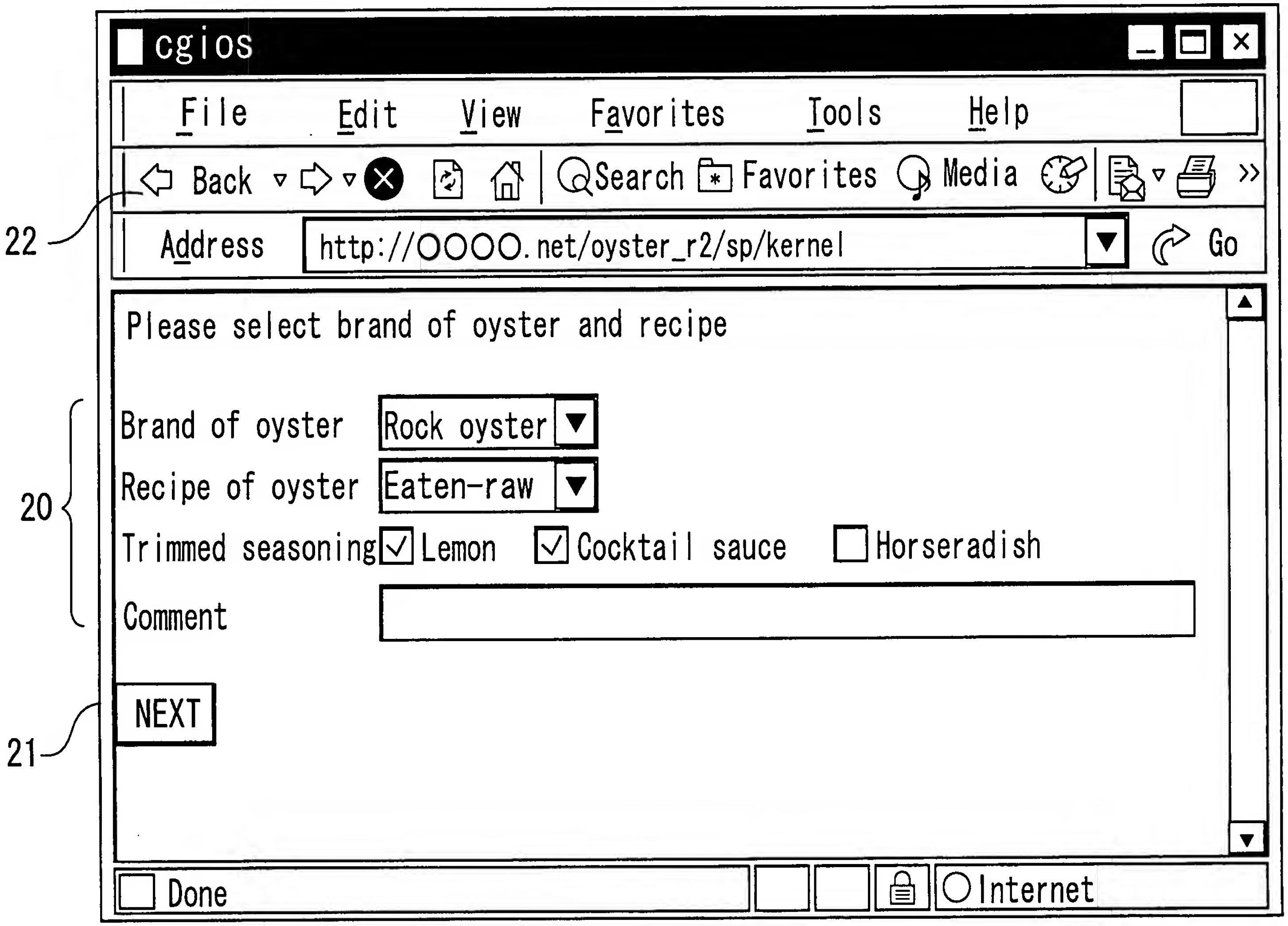
Recipe of oyster: Eaten-raw ▼

Trimmed seasoning ☑ Lemon □ Cocktail sauce □ Horseradish

Comment

10/538805

FIG.9



http://OOOO.net/oyster_r2/sp/kernel

Please select brand of oyster and recipe

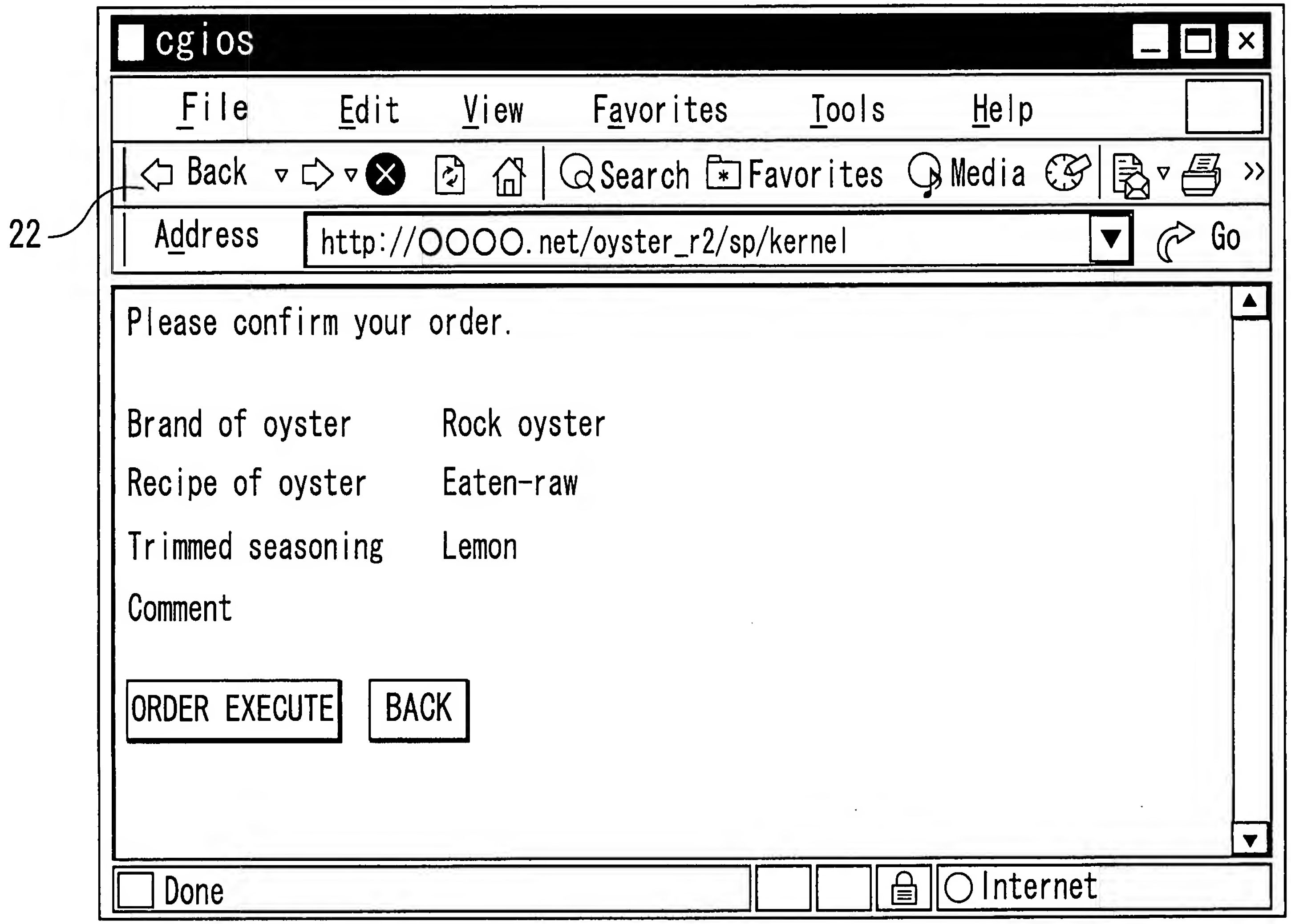
Brand of oyster Rock oyster ▼

Recipe of oyster Eaten-raw ▼

Trimmed seasoning ☑ Lemon ☑ Cocktail sauce □ Horseradish

Comment

FIG.10



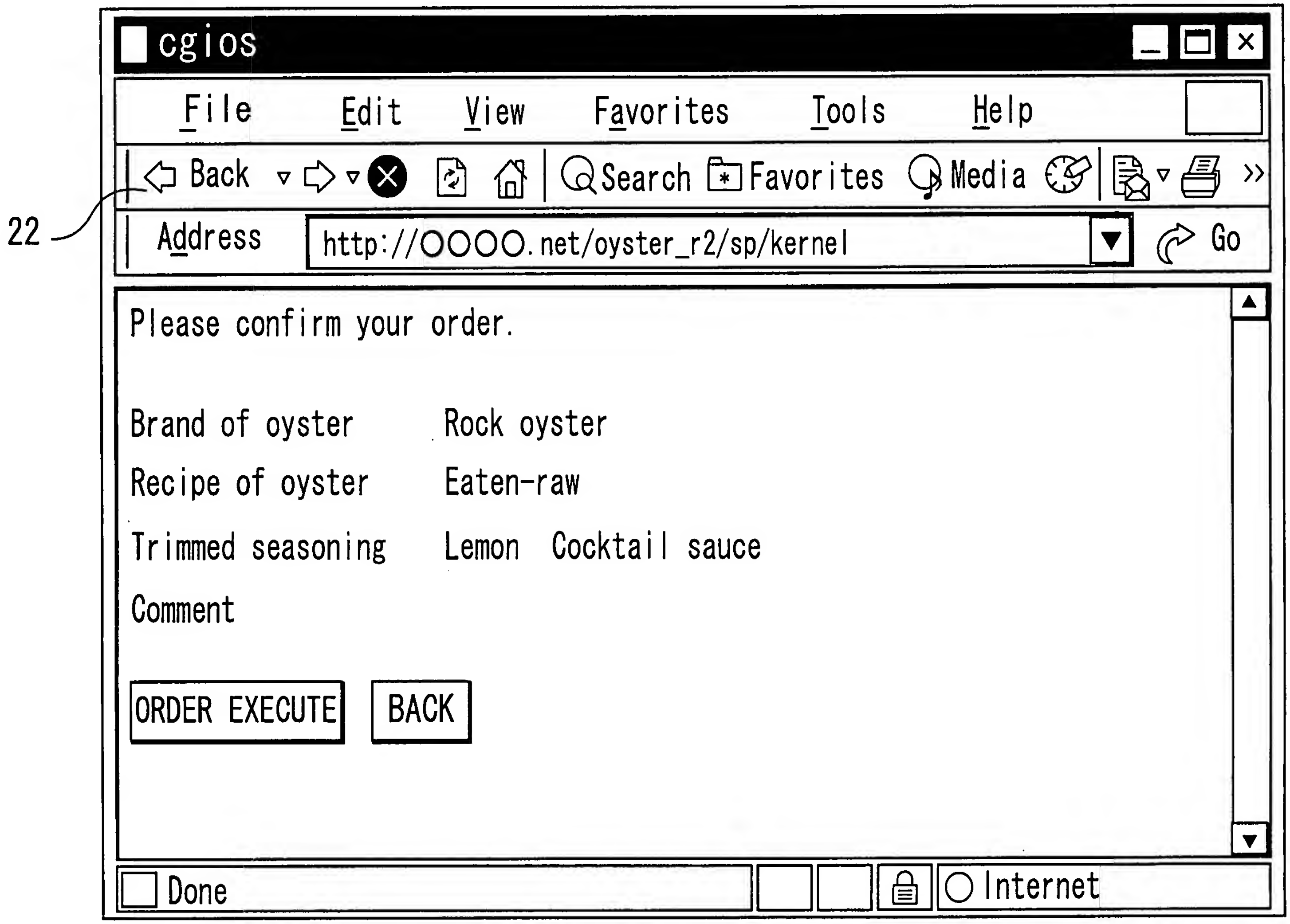
http://○○○○.net/oyster_r2/sp/kernel

Please confirm your order.

Brand of oyster	Rock oyster
Recipe of oyster	Eaten-raw
Trimmed seasoning	Lemon
Comment	

ORDER EXECUTE　BACK

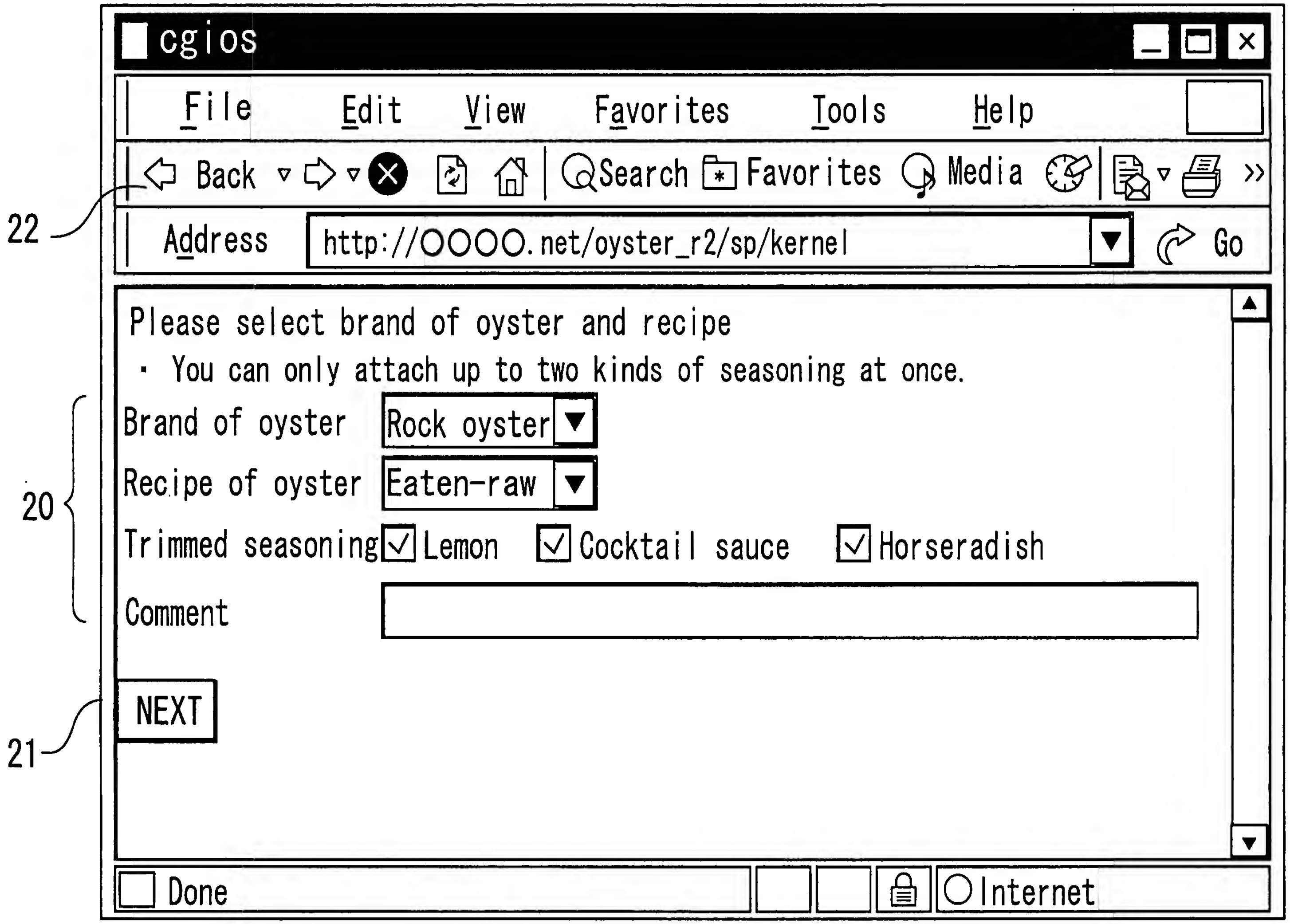
FIG.11



Please confirm your order.

Brand of oyster	Rock oyster
Recipe of oyster	Eaten-raw
Trimmed seasoning	Lemon Cocktail sauce
Comment	

FIG.12



Please select brand of oyster and recipe

· You can only attach up to two kinds of seasoning at once.

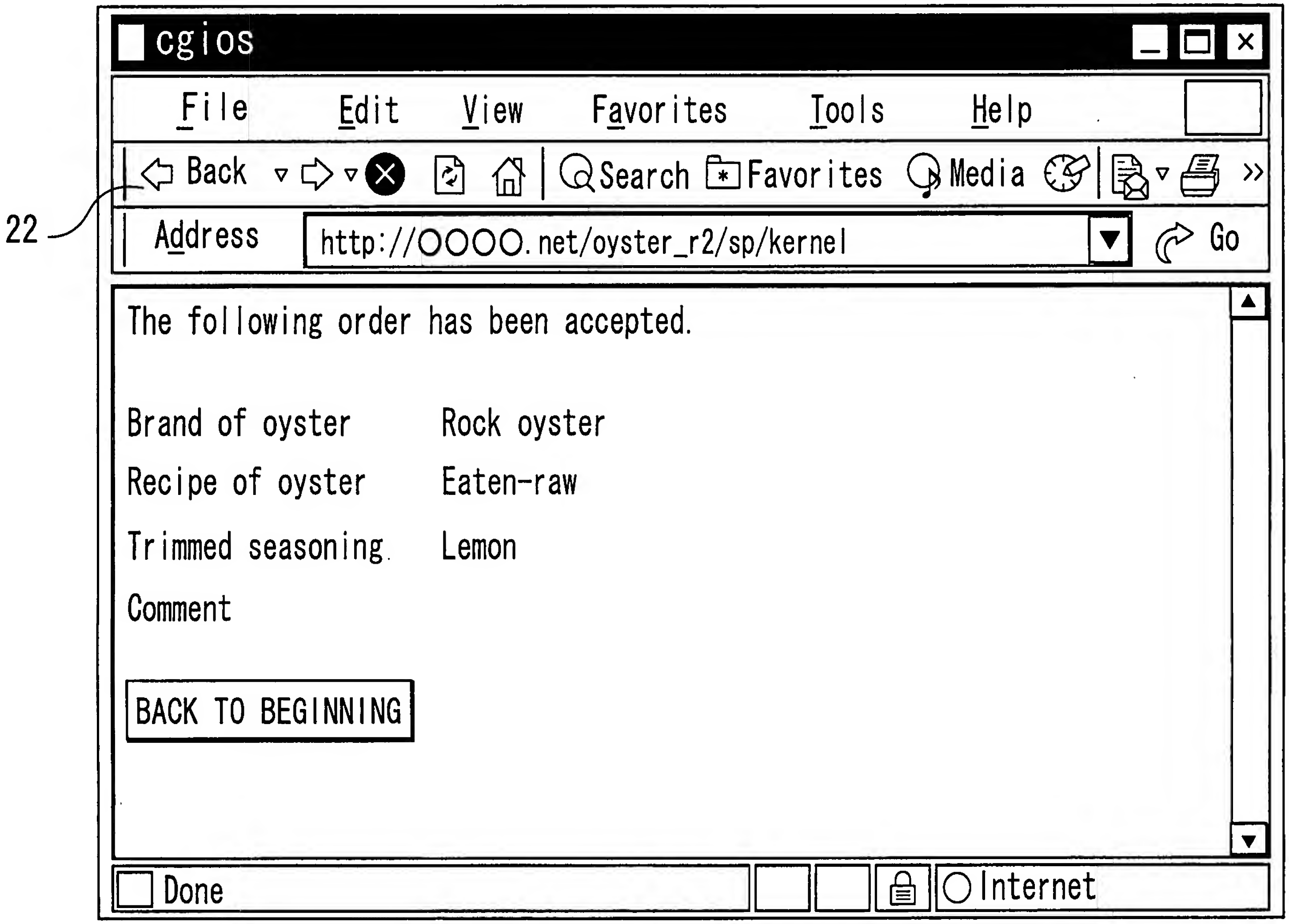
Brand of oyster: Rock oyster ▼

Recipe of oyster: Eaten-raw ▼

Trimmed seasoning ☑ Lemon ☑ Cocktail sauce ☑ Horseradish

Comment

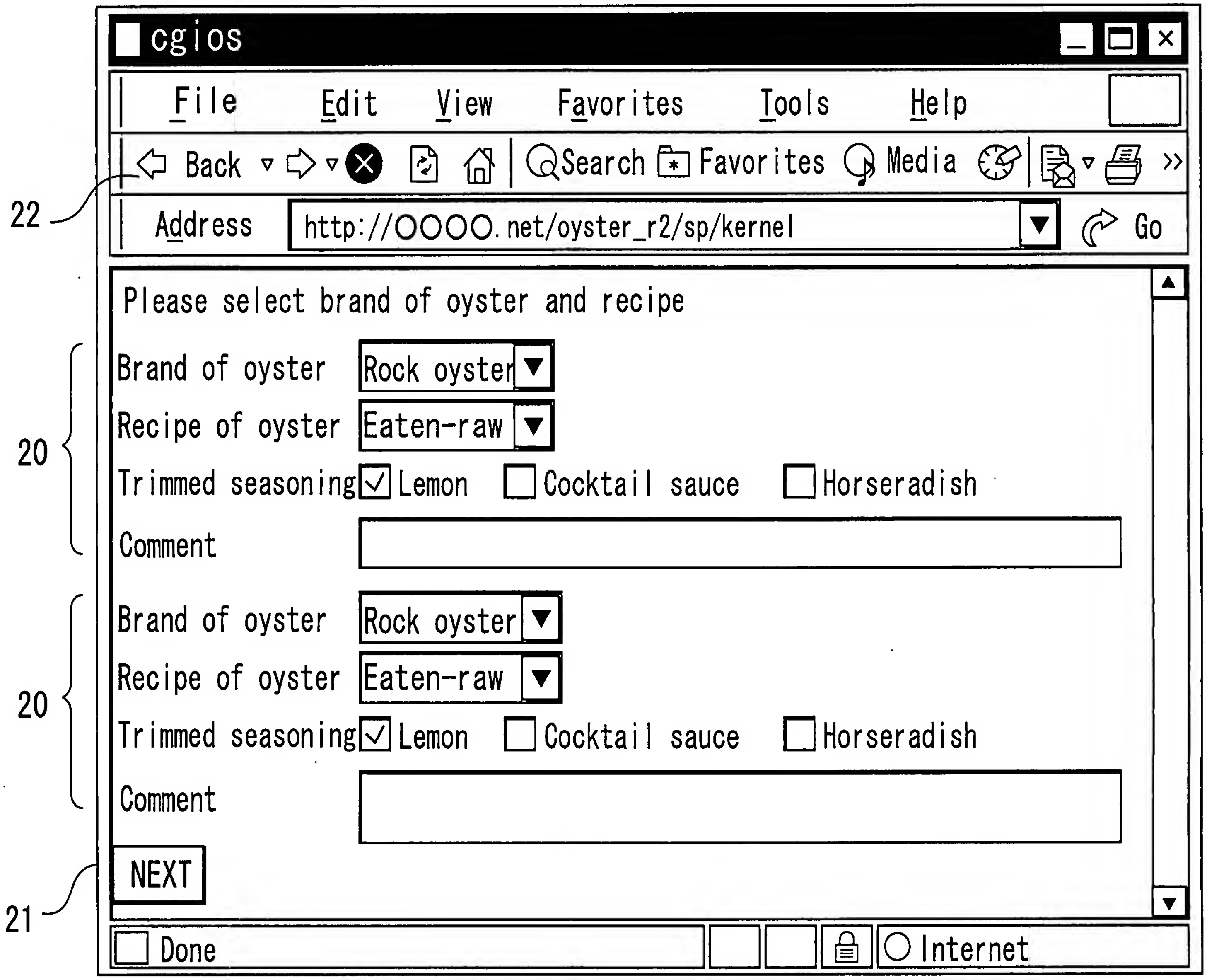
FIG.13



The following order has been accepted.

Brand of oyster	Rock oyster
Recipe of oyster	Eaten-raw
Trimmed seasoning	Lemon
Comment	

FIG.14



http://OOOO.net/oyster_r2/sp/kernel

Please select brand of oyster and recipe

Brand of oyster Rock oyster ▼

Recipe of oyster Eaten-raw ▼

Trimmed seasoning ☑ Lemon ☐ Cocktail sauce ☐ Horseradish

Comment

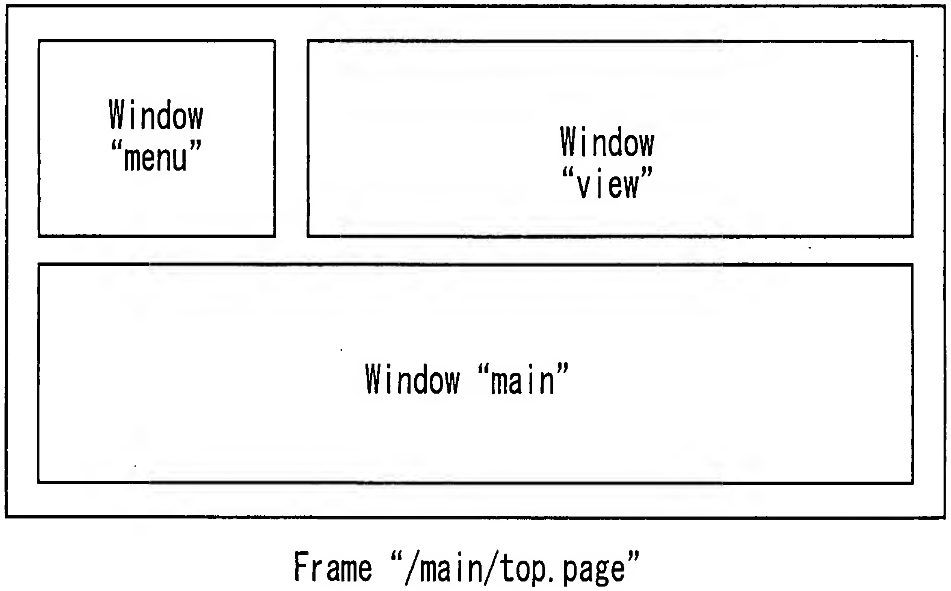
Brand of oyster Rock oyster ▼

Recipe of oyster Eaten-raw ▼

Trimmed seasoning ☑ Lemon ☐ Cocktail sauce ☐ Horseradish

Comment

FIG.15



10/538805

FIG.16

```
(a) <Activation stream:/init.stream>
        ustream init() { # Activation stream
         loop:
          p_fork("Window stream","/main/init.stream","this");
          p_wait();
          goto loop;
}
```

```
(b) <Window activation stream:/main/init.stream>
        ustream init() { # Window stream
          loop:
            p_setPage("top.page");
            p_fork("menu","menu.stream","menu");
            p_wait();
            goto loop;
}
```

```
(c) <Menu stream:/main/menu.stream>
        ustream menu() { # Menu display stream
          first:
            page(menu.page) {
            }
            if ($V->{action} eq "input_order") {
                p_fork("INPUT ORDER","input_order.stream","^view");
            } elsif ($V->{action} eq "list") {
                p_fork("DISPLAY ORDERING LIST","list.stream","^main");
            }
            goto first;
}
```

FIG.17

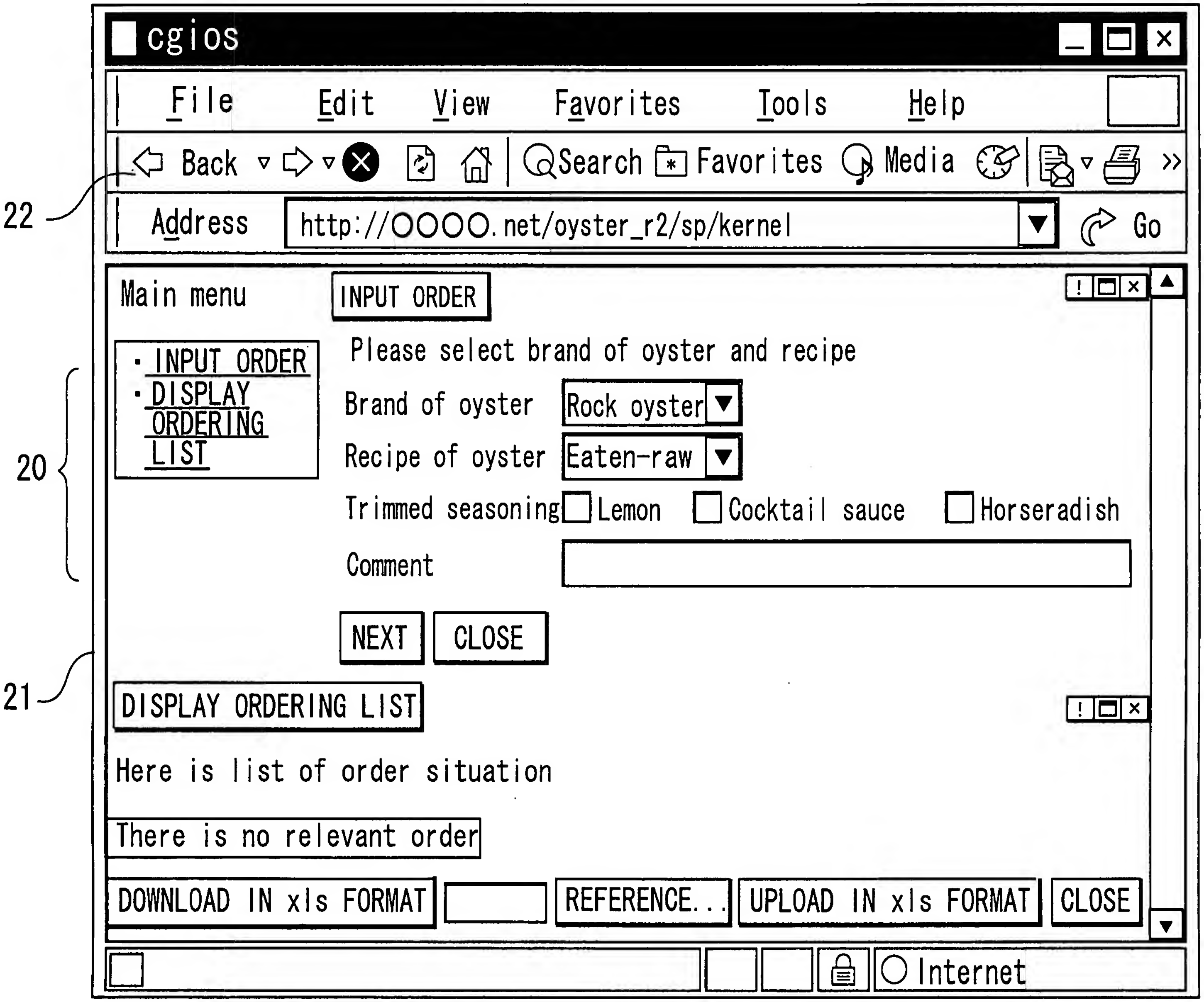
(a) <Top frame (new addition):/main/top.page>

```
        <table width=100% height=100% bgcolor=#ffffff border=0 cellspacing=2
cellpadding=0 style="table-layout:fixed">
         <tr>
          <td width=143 valign=top align=left class=wallpaper>
           <div style="overflow:auto"><#menu></div></td>
          <td valign=top class=wallpaper>
           <div style="overflow:auto"><#view></div></td>
         </tr>
         <tr>
          <td colspan=2 height=70% valign=top class=wallpaper>
           <div style="overflow:auto"><#main></div></td>
         </tr>
</table>
```

(b) <Menu page (new addition):/main/menu.page>

```
        <h3> Main menu </h3>
        <table><tr><td valign=top>
        <table class=inner bgcolor=#ffffff border-color=#000000 border=2
cellspacing=0 cellpadding=10>
        <tr><td valign=top>
        <li><a href="action:input_order"> INPUT ORDER</a>
        <li><a href="action:list"> DISPLAY ORDERING LIST</a>
        </td></tr>
        </table>
</td></tr></table>
```

FIG.18



http://OOOO.net/oyster_r2/sp/kernel

Please select brand of oyster and recipe

Brand of oyster: Rock oyster

Recipe of oyster: Eaten-raw

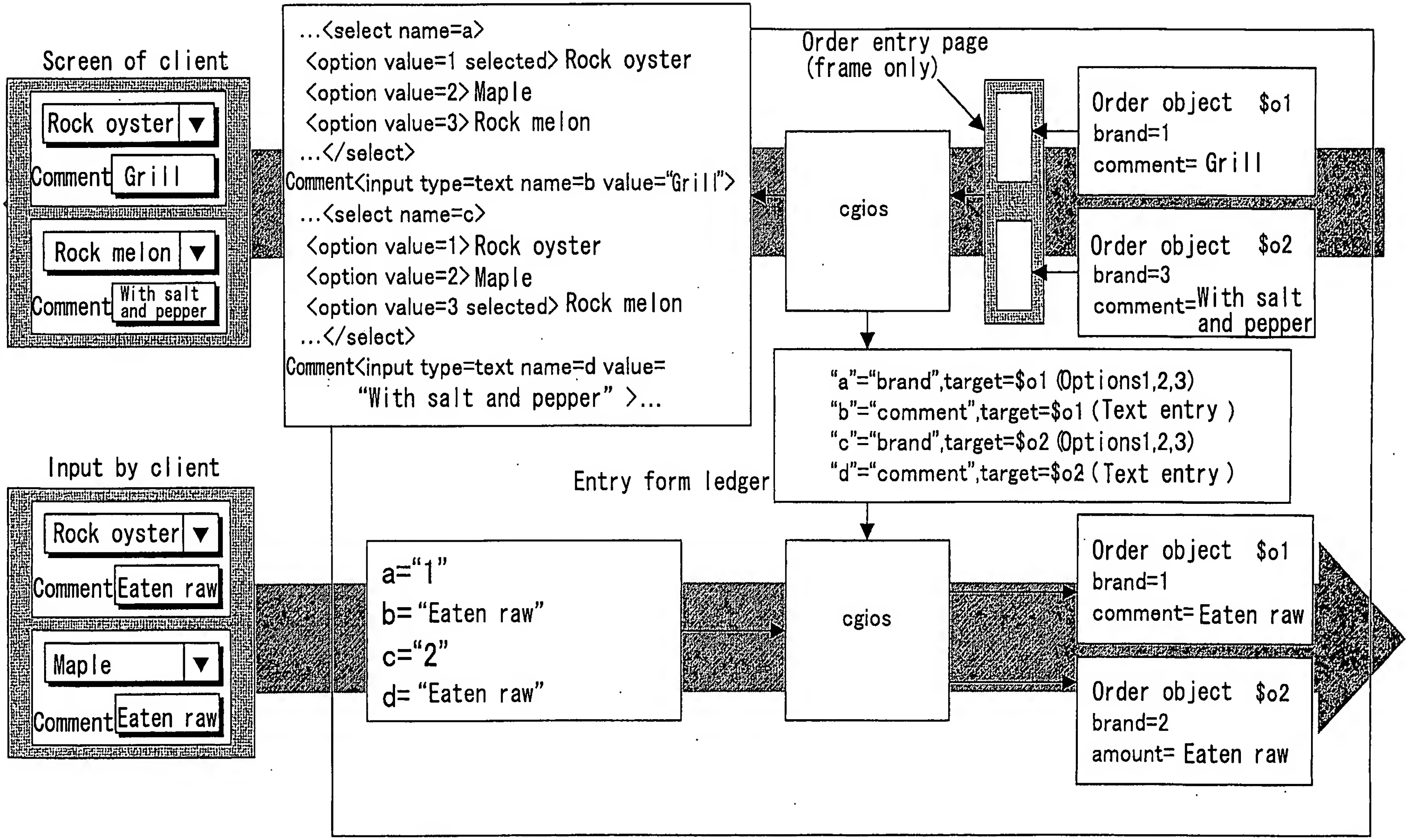
Trimmed seasoning: □ Lemon □ Cocktail sauce □ Horseradish

Comment

Here is list of order situation

There is no relevant order

FIG.19

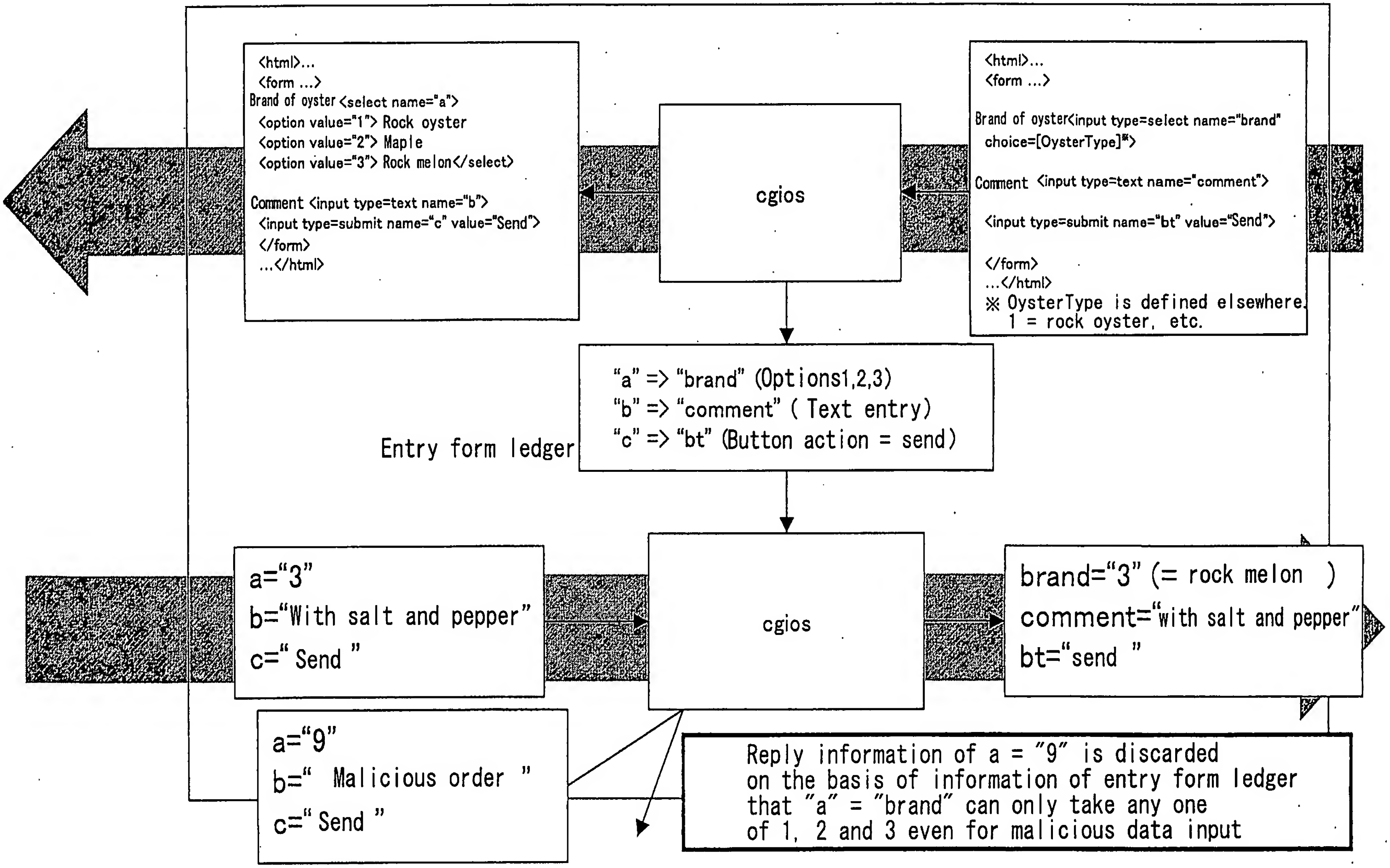


```
...<select name=a>
 <option value=1 selected> Rock oyster
 <option value=2> Maple
 <option value=3> Rock melon
 ...</select>
Comment<input type=text name=b value="Grill">
 ...<select name=c>
 <option value=1> Rock oyster
 <option value=2> Maple
 <option value=3 selected> Rock melon
 ...</select>
Comment<input type=text name=d value=
     "With salt and pepper" >...
```

```
"a"="brand",target=$o1 (Options1,2,3)
"b"="comment",target=$o1 (Text entry )
"c"="brand",target=$o2 (Options1,2,3)
"d"="comment",target=$o2 (Text entry )
```

```
a="1"
b= "Eaten raw"
c="2"
d= "Eaten raw"
```

FIG.20



```
<html>...
<form ...>
Brand of oyster <select name="a">
 <option value="1"> Rock oyster
 <option value="2"> Maple
 <option value="3"> Rock melon</select>

Comment <input type=text name="b">
 <input type=submit name="c" value="Send">
 </form>
 ...</html>
```

```
<html>...
<form ...>

Brand of oyster<input type=select name="brand"
 choice=[OysterType]※>

Comment <input type=text name="comment">

<input type=submit name="bt" value="Send">

</form>
...</html>
※ OysterType is defined elsewhere.
  1 = rock oyster, etc.
```

"a" => "brand" (Options1,2,3)
"b" => "comment" (Text entry)
"c" => "bt" (Button action = send)

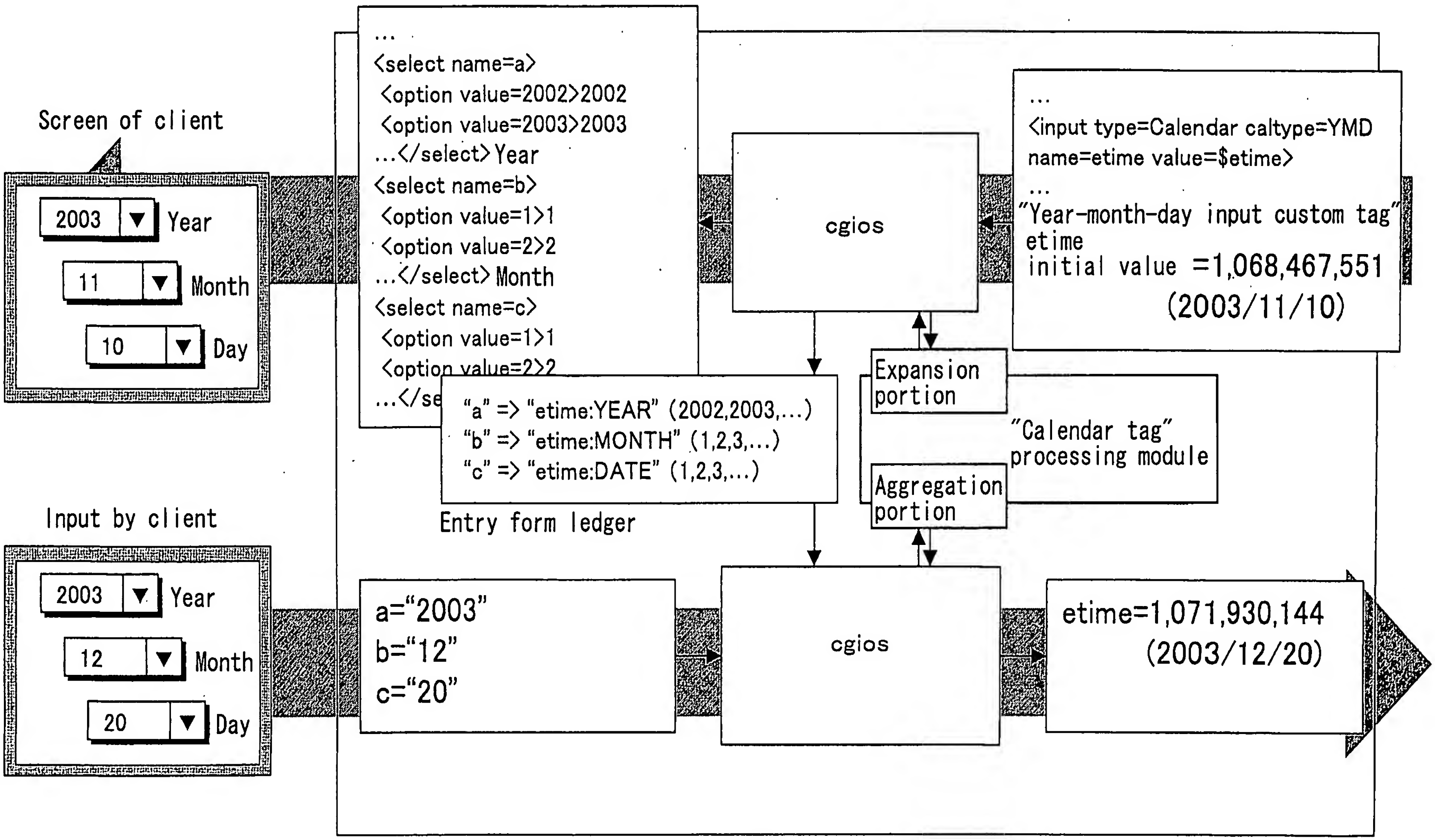
a="3"
b="With salt and pepper"
c=" Send "

brand="3" (= rock melon)
comment="with salt and pepper"
bt="send "

a="9"
b=" Malicious order "
c=" Send "

Reply information of a = "9" is discarded on the basis of information of entry form ledger that "a" = "brand" can only take any one of 1, 2 and 3 even for malicious data input

FIG.21



```
...
<select name=a>
 <option value=2002>2002
 <option value=2003>2003
...</select>Year
<select name=b>
 <option value=1>1
 <option value=2>2
...</select>Month
<select name=c>
 <option value=1>1
 <option value=2>2
...</se
```

```
"a" => "etime:YEAR" (2002,2003,...)
"b" => "etime:MONTH" (1,2,3,...)
"c" => "etime:DATE" (1,2,3,...)
```

```
...
<input type=Calendar caltype=YMD
name=etime value=$etime>
...
"Year-month-day input custom tag"
etime
initial value =1,068,467,551
              (2003/11/10)
```

```
a="2003"
b="12"
c="20"
```

```
etime=1,071,930,144
      (2003/12/20)
```

FIG.22

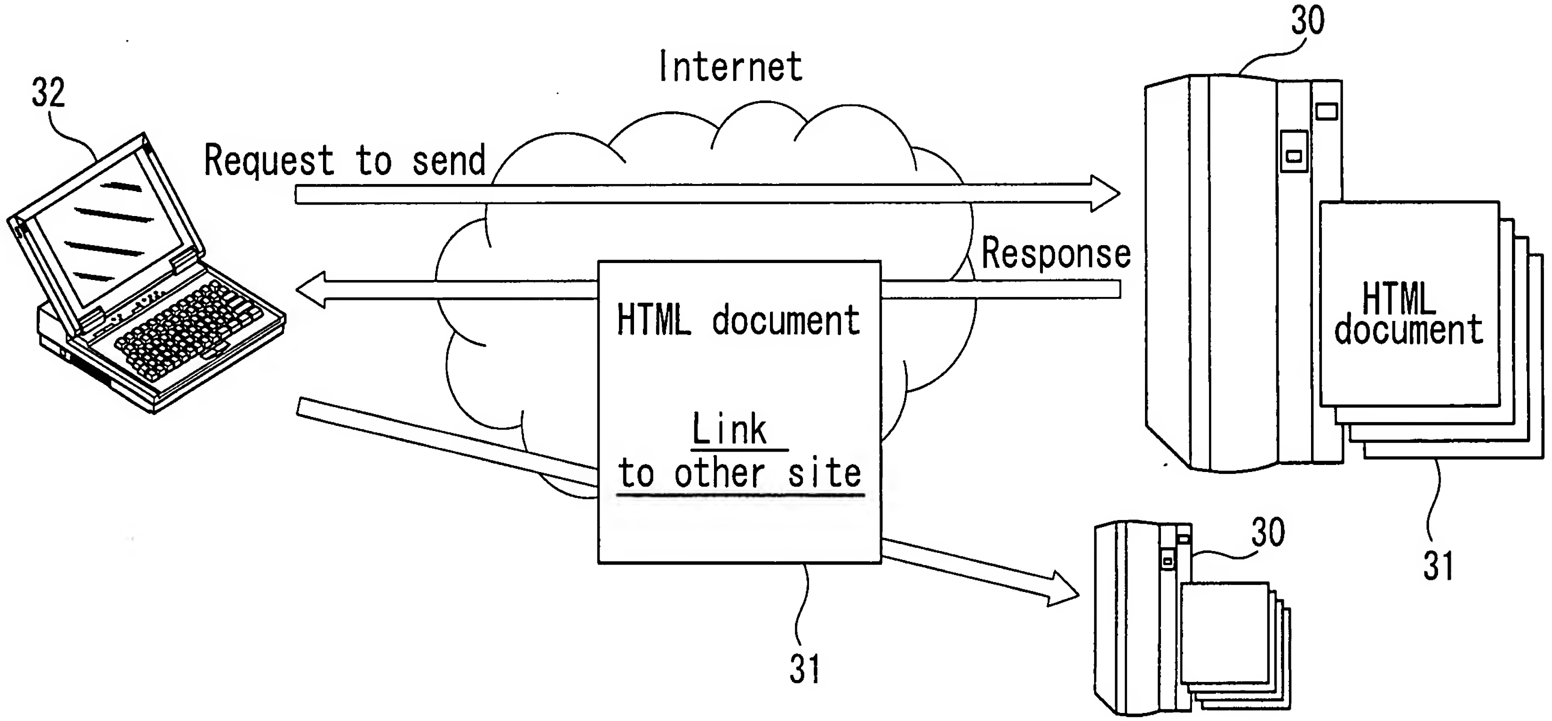

32
Request to send
Internet
Response
30
HTML document
HTML document
Link to other site
31
30
31

FIG.23

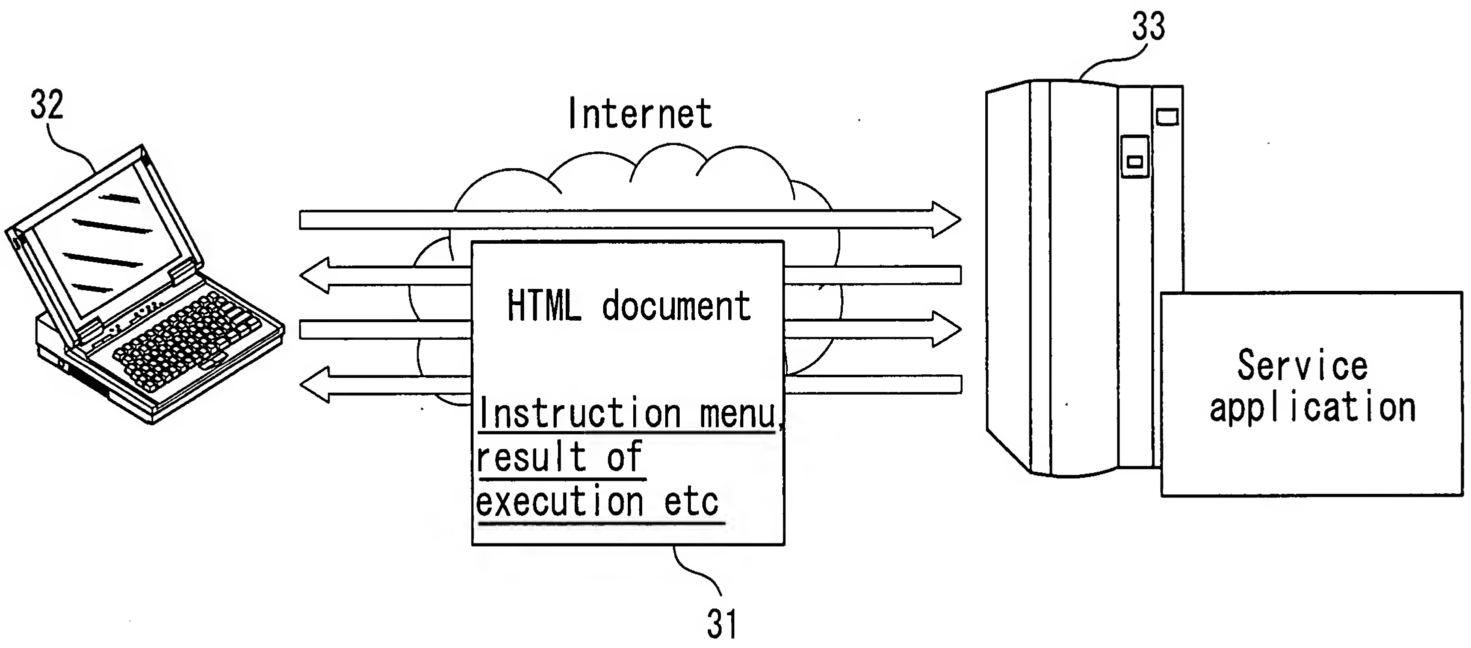

32
Internet
33
HTML document
Instruction menu,
result of
execution etc
31
Service
application

FIG.24

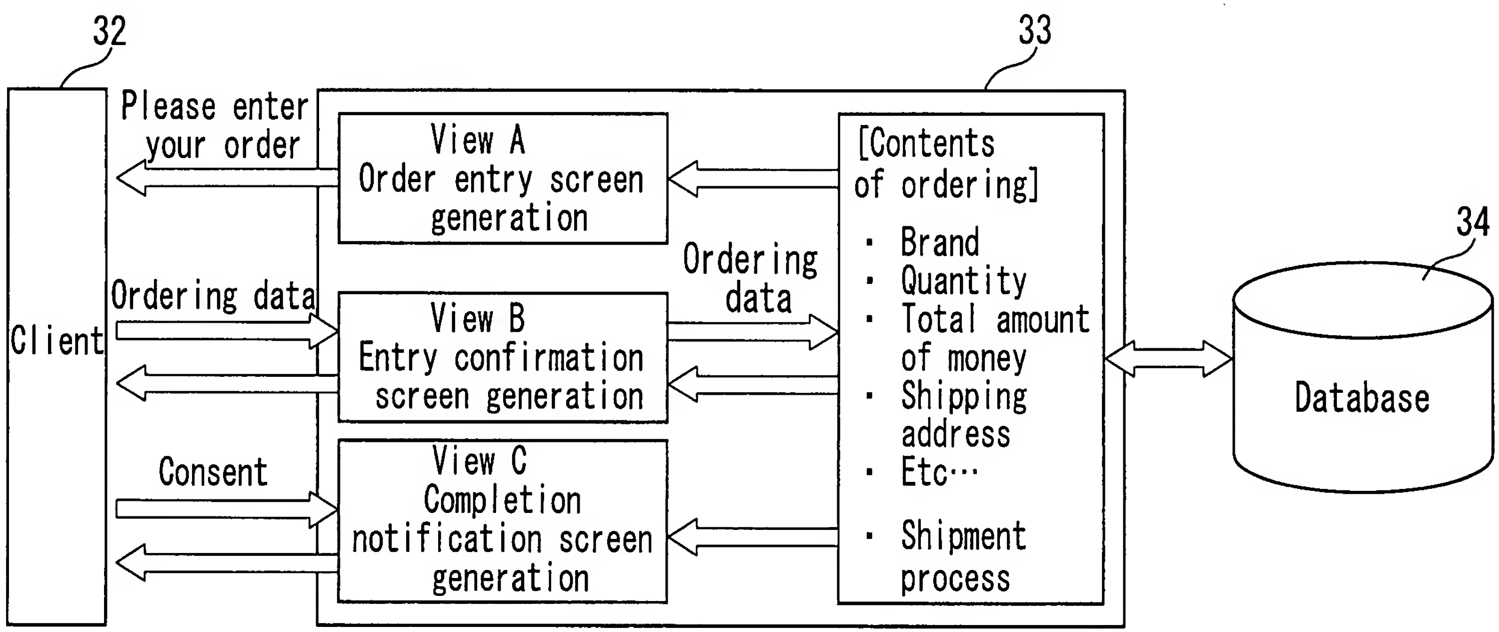

32
33
34
Client
Please enter your order
Ordering data
Consent
View A
Order entry screen generation
View B
Entry confirmation screen generation
View C
Completion notification screen generation
Ordering data
[Contents of ordering]
· Brand
· Quantity
· Total amount of money
· Shipping address
· Etc…
· Shipment process
Database

FIG.25

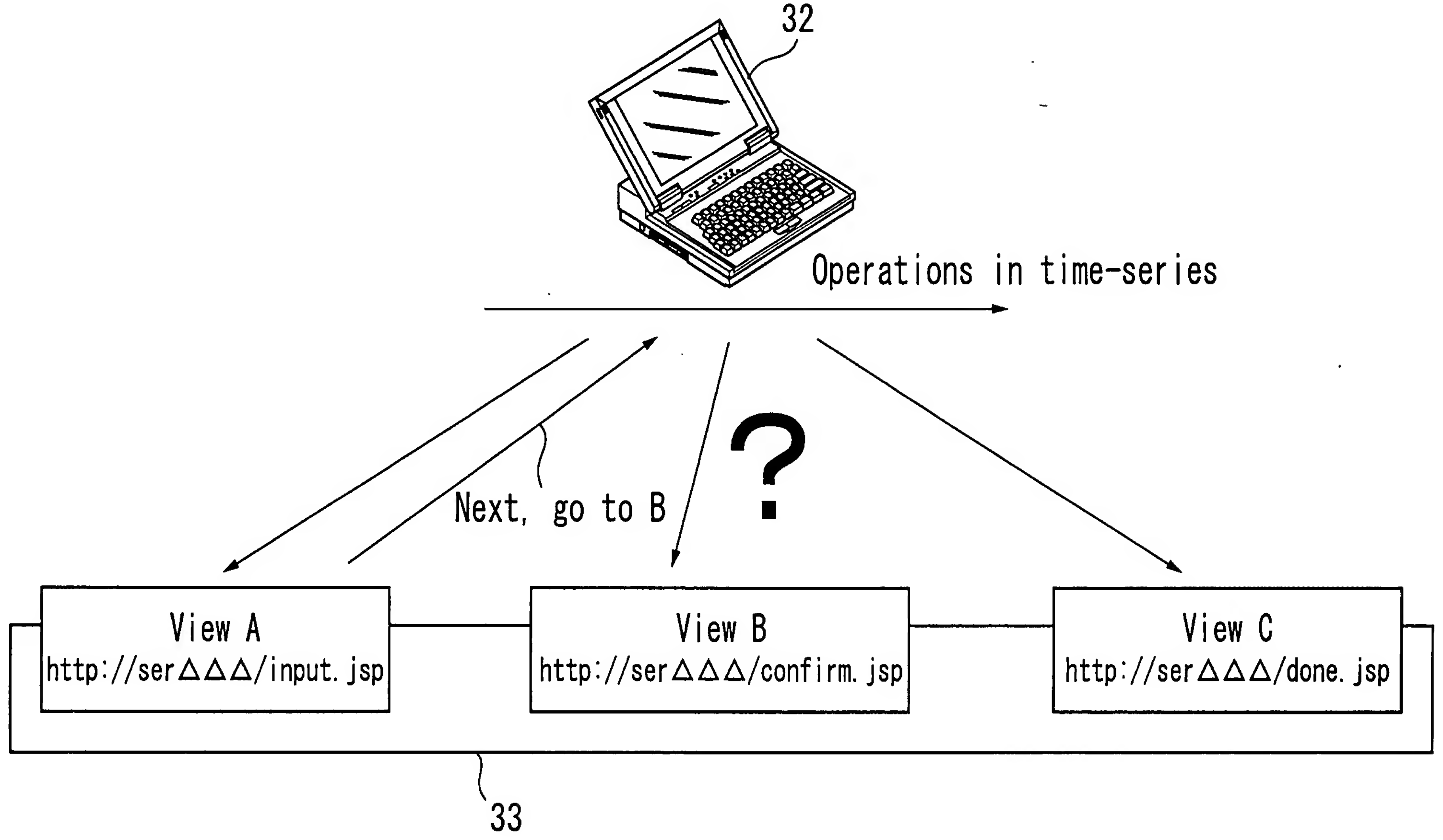

32
Operations in time-series
Next, go to B
?
View A
http://ser△△△/input. jsp
View B
http://ser△△△/confirm. jsp
View C
http://ser△△△/done. jsp
33

FIG.26

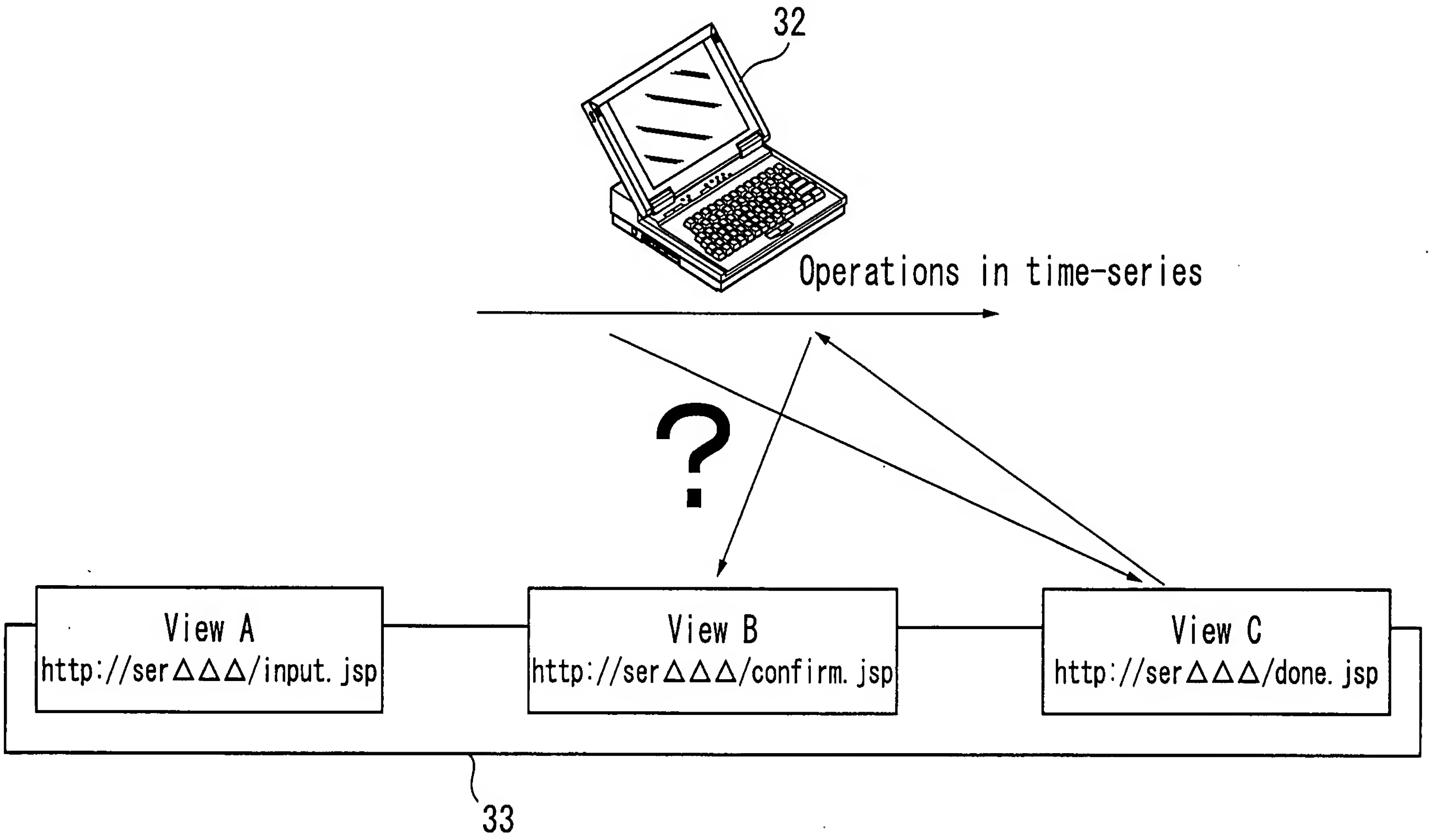
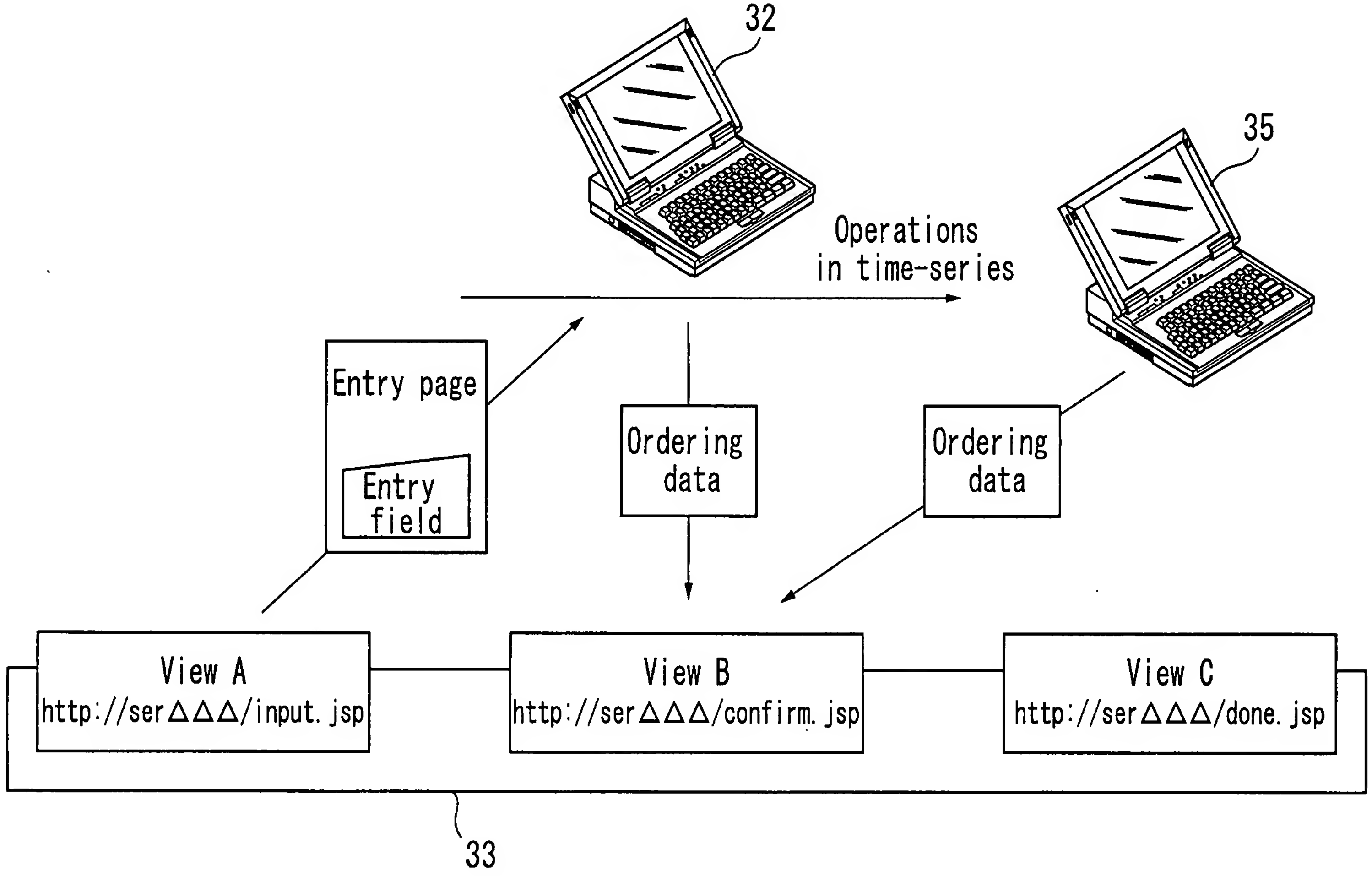

32
Operations in time-series
?
View A
http://ser△△△/input. jsp
View B
http://ser△△△/confirm. jsp
View C
http://ser△△△/done. jsp
33

FIG.27


32
35
Operations in time-series
Entry page
Entry field
Ordering data
Ordering data
View A
http://ser△△△/input. jsp
View B
http://ser△△△/confirm. jsp
View C
http://ser△△△/done. jsp
33